Fast Approximated SIFT*

Michael Grabner, Helmut Grabner, and Horst Bischof

Institute for Computer Graphics and Vision,
Graz University of Technology, Austria
{mgrabner, hgrabner, bischof}@icg.tu-graz.ac.at

Abstract. We propose a considerably faster approximation of the well known SIFT method. The main idea is to use efficient data structures for both, the detector and the descriptor. The detection of interest regions is considerably speed-up by using an integral image for scale space computation. The descriptor which is based on orientation histograms, is accelerated by the use of an integral orientation histogram. We present an analysis of the computational costs comparing both parts of our approach to the conventional method. Extensive experiments show a speed-up by a factor of eight while the matching and repeatability performance is decreased only slightly.

1 Introduction

In the last few years we have witnessed an explosion of object recognition methods based on the detection of local key-points and construction of local photometric descriptors around these key-points (e.g. [1, 2, 3, 4]). The basic idea of these approaches is to first detect salient structures in images (e.g., corners, high entropy regions, scale space maxima, etc.) and to construct from the region or its surrounding a discriminative description which is used for matching. The requirement is that the structures can be re-detected with high reliability and that the descriptor is robust (e.g. to illumination changes) and possesses certain invariance properties (e.g. affinely invariant). The big advantage of these approaches is that they do not require a segmentation of the image and due to the local nature they are robust to occlusions.

Local approaches have demonstrated considerable success in a variety of applications, like recognition of objects [1], wide-base line stereo [4], robot navigation [5], image retrieval [6, 7], building of panoramas [8], etc. Probably the most popular and widely used local approach is the DoG detector with the SIFT descriptor as proposed by Lowe [1]. SIFT has been used with success in all of the above mentioned application areas. Evaluations and comparison

* The project results have been developed in the MISTRAL Project which is financed by the Austrian Research Promotion Agency (www.ffg.at). This work has been sponsored in part by the Austrian Federal Ministry of Transport, Innovation and Technology under P-Nr. I2-2-26p VITUS2 and by the Austrian Joint Research Project Cognitive Vision under projects S9103-N04 and S9104-N04, the EC funded NOE MUSCLE IST 507572

P.J. Narayanan et al. (Eds.): ACCV 2006, LNCS 3851, pp. 918–927, 2006.

(e.g. [9]) demonstrate the excellent performance of the method compared to other approaches. The DoG detector detects blobs in the Laplacian scale space. The SIFT descriptor is basically a histogram (in fact 16 concatenated ones) of gradient orientations of the normalized (with respect to scale and orientation) DoG region. One key issue for its success is that DoG points and SIFT are normalized with each other and can be computed fast.

Due to the high popularity of SIFT, it is no surprise that several variants and extensions of SIFT have been proposed. For example Ke and Sukthankar proposed the so called PCA-SIFT [10] that applies Principal Components Analysis (PCA) to the normalized gradient patch. The Gradient location and orientation histogram (GLOH) [9] changes SIFTs location grid and uses PCA to reduce the size of SIFT. The primary focus of these extensions is to gain improved performance.

In this paper we propose a modified SIFT method for recognition purpose. Our primary motivation is to significantly speed up the SIFT computation while at the same time keep the excellent matching performance. We demonstrate that by using approximations (mainly employing integral images) both the DoG detector (see section 2) and the SIFT-descriptor (see section 3) we can speed-up the SIFT computation by at least a factor of eight compared to the binaries provided by Lowe. Extensive experimental evaluations (see section 4) show that the loss in matching performance is negligible.

2 DoG Detector

In order to detect scale invariant key-points Lowe suggests to repeatedly smooth the input image and identify key locations in scale space. In order to detect even very small scales Lowe extends this approach and proposes to double the input image before building the scale space. The different scale levels are produced by recursive filtering with a variable-scale Gaussian kernel. A local maxima search is finally applied to the Difference-of-Gaussian images which can be computed of adjacent scale images, in order to detect key-points in scale space.

To accelerate this approach we propose several approximations and changes, see Table 1. The key idea of our method is to considerably reduce the costs for computing the scale space by using Difference-of-Mean (DoM) images instead of Difference-of-Gaussians (DoG). This DoM images can be computed very efficiently by using a box filter in combination with an *integral image* as introduced

Table 1. Major differences between Lowe's detector [1] and our proposed approach

SIFT	Fast approximated SIFT
image doubling	-
-	calculate integral image
DoG scale space	DoM scale space
post-processing	-

by Viola and Jones [11] (capturing the main idea of [12]). Once the integral image is computed, it allows to compute the mean within a rectangular region in constant time independent of the size of the region. This property allows fast box filtering and can be used for linear sampling of the scale axis which is realized by successively increasing the size of the filter kernel. Adjacent scale space images are subtracted and a local maxima search is applied to the Difference-of-Mean images in order to detect key-points. For a reliable detection of key-points at all scales it is important to normalize the DoM response with

$$sensitivity \cdot \left(1 - \frac{s_1^2}{s_2^2}\right) \tag{1}$$

where s_1, s_2 corresponds to the size of the small and larger box filter, respectively. The parameter *sensitivity* captures the minimal contrast of the mean gray values of the inner region (s_1) and the outer region ($s_2 - s_1$) and can be used to adjust the sensitivity of the detector. Since experiments with DoG indicate that small scales cannot be reliably matched we skip the doubling of the image size, which again provides a significant speed-up. Once the key-points have been detected we do not make any further post-processing like an accurate key-point localization because due to the use of integral images we have already pixel accuracy at each scale. But note that the accuracy of the obtained points is not as precise as with the DoG, nevertheless the detected points are good for recognition tasks but less suitable for geometric tasks like estimation of the fundamental matrix.

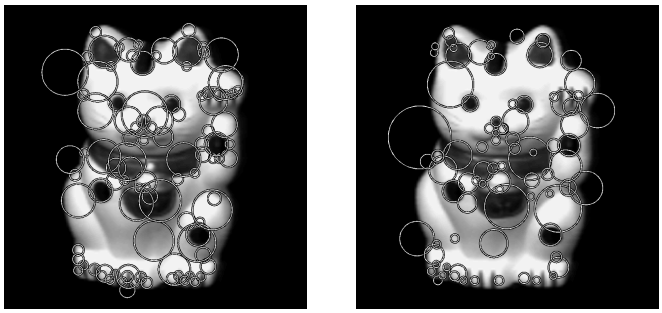


Fig. 1. Comparison of the DoM key-points (left) detected by our approach to DoG key-points (right) detected by the approach of Lowe

2.1 Computational Costs

The box filtering approach using integral images is depicted in Algorithm 1. Once the integral image is pre-computed which takes 2 additions for each image pixel, a single box filter response can be computed, independent of its size, with 4 memory accesses, 3 additions and a single multiplication which is needed for normalizing the box region. In Table 2 which has been adapted from [13], we compare the box filtering approach to other commonly used Gaussian filtering

Algorithm 1. Integral image computation

```
// pre-computation
for each image point do
  Propagate integral image {1 addition}
  Increase value {1 addition}
end for
// apply box filter with a given kernel size
for each image point do
  Compute intersection {3 addition}
  Normalize {1 multiplication}
end for
```

Table 2. Comparison of various filtering techniques (calculations per pixel)

Filter technique	Additions	Multiplications
2D-Gauss	N^2	$N^2 - 1$
Separated Gauss	$2 \cdot N - 2$	$N + 2$
Recursive Gauss	6	14
FFT	$2 \cdot log(W \cdot H)$	$2 \cdot log(W \cdot H) + 1$
Box filter	$2 + 3$	1

techniques. Simple 2-D convolution is the slowest one since the complexity for each pixel is $O(N^2)$, where N corresponds to the filter size. Much more efficient is to make use of the separability of the Gaussian function which allows convolution by applying two passes of the 1-D function in the horizontal and vertical directions. This leads to linear costs in the kernel size N. Other methods like FFT are independent with respect to the filter kernel size but depend on the size of the input image $W \times H$. However, as can be seen in Figure 2(a), the computational costs are higher than for the separable Gaussian for a kernel size of 7×7 (as proposed by Lowe in [14]). A similar result holds for recursive Gaussian filters which allow convolution in constant time but are still computationally more demanding for small filter kernels.

3 SIFT Descriptor

Reliable matching of key-points is performed by feature vectors generated from their local neighborhoods. Lowe suggests to use the gradient information around a key-point. Initially a consistent orientation is assigned to the key-point such that the descriptor can be represented relative to this orientation, thereby achieving rotation invariance. Gradients within a circular region are used to compute an orientation histogram, and local maxima in the histogram are used as characteristic orientations.

To obtain a descriptor Lowe proposes to divide the surrounding region into 4×4 sub-patches. From each sub-patch an orientation histogram with 8 bins

is computed and concatenated to form a single feature vector. Since orientation histograms form the basic computation for the descriptor this leads to the idea to use integral histograms [15]. Integral histograms are an extension of integral images using for each histogram bin (e.g. orientation) a separate integral image. Once the integral orientation histogram is computed, histograms can be accessed in constant time independent of the size of the region. Similar to integral images integral histograms can only provide histograms of rectangular regions.

For orientation histogram computation we use un-weighted squared regions. Furthermore, for the descriptor we rotate the midpoints of each sub-patch relative to the orientation and compute the histograms of overlapping sub-patches without aligning the squared region but shifting the sub-patch histogram relative to the main orientation. The main advantage of our method is that we make use of the full resolution of the input image without additional computational costs.

3.1 Computational Costs

The major question is how many descriptors have to be calculated in order to obtain a speed up for the integral version compared to the conventional approach. We define the costs for single histogram computation for both approaches which has been done by adapting the analysis from [15]. We assume that the gradient image has already been computed. In addition we assume computing histograms only over squared regions.

Algorithm 2. Conventional histogram computation

```
//histogram computation
for each histogram do
  for each gradient within window do
    Find bin { 1 multiplication}
    Increase bin value { 1 addition}
  end for
end for
```

The conventional method for histogram computation is given in Algorithm 2. Once the gradient image is available, for each gradient in the observed region an assignment to the correct bin value must be done. Consequently the conventional method strongly depends on the number of gradients contributing to the histogram which leads to the complexity $O(N^2)$ for a squared region where N corresponds to the window size. In addition the computational costs for a squared region is

$$k \cdot N^2 \cdot (c_{add} + c_{mult}) \tag{2}$$

where k corresponds to the number of histograms, c_{add} represent costs for an addition and c_{mult} are the costs for a multiplication.

Considering the integral histogram computation illustrated in Algorithm 3, we see that equivalent to integral images some pre-computations have to be

Algorithm 3. Integral histogram computation

```
//pre-computation
for each gradient do
  for each bin do
    Propagate integral histogram { 1 addition}
  end for
  Find bin { 1 multiplication}
  Increase bin value { 1 addition}
end for
//histogram computation
for each histogram do
  for each bin do
    Compute intersection { 3 additions}
  end for
end for
```

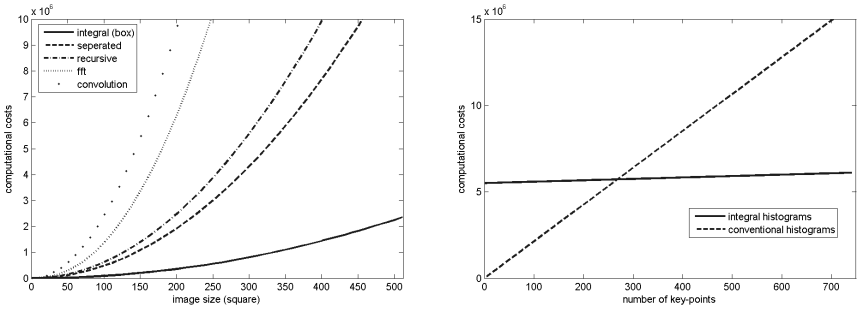
done. Once the integral orientation histogram has been computed, orientation histograms can accessed in $k \cdot b \cdot 3 \cdot c_{add}$, where b corresponds to the number of bins (in our case 16 bins are used). Similar to integral images rectangular regions can be accessed. The costs for histogram computation does not depend on the number of gradients within a region.

Consequently the total costs including the computation of the integral orientation histogram can be written as

$$W \cdot H \cdot (b \cdot c_{add} + c_{add} + c_{mult}) + k \cdot b \cdot 3 \cdot c_{add} \tag{3}$$

where $W \times H$ represents the input image size.

Figure 2(b) compares standard histogram and integral histogram computation, where we have used relative costs for additions and multiplications from [15]



(a) Different filtering techniques for a 7×7 filter kernel

(b) Conventional and integral technique for orientation histogram computation

Fig. 2. Comparison of computational costs for detector (left) and the descriptor (right)

(addition:1 - multiplication:4). Other parameters of the cost functions, such as the histogram patch size, have been experimentally determined. As we can see in Figure 2(b), initially the costs for the integral histogram are much higher however once the integral image is computed the costs increase very slowly. In contrast the costs of the conventional method increase linearly with the number of computed descriptors.

Integral orientation histograms are profitable especially when calculated over large regions. This is especially suited for our approach because we always compute the descriptors on the original resolution. Consequently, we take advantage of using the whole information of the input image.

4 Experimental Results

We compare our novel approach to Lowe's method with respect to performance and speed. For matching performance we run two types of experiments to explore the effects of the approximations made in our approach. First, both methods are examined with respect to rotation, scale and perspective invariance on a data-set of 15 commonly used images. Second, an evaluation comparing both, detectors and descriptors, on 2 images of the popular Graffiti data-set has been done using the framework of Mikolajczyk [9]. Finally we compare the runtime of our approach to Lowe's publicly available binaries [1].

4.1 Artificial Transformations

For all artificial transformations we used the same criterions for determining repeatability of the detector and the matching score of the descriptor. The repeatability is obtained through a simple location criterion while for the matching score a key-point match and the corresponding nearest descriptor match is required.

Due to the box filter approximation the rotation is the worst case scenario for the detector. Even for the descriptor the worst case because no rotational sampling is done. Therefore we artificially rotate each image from 0°to 90°of our data-set with steps of 15°. In Figure 3 we see that both, the detector and the descriptor of the approximated SIFT implementation behave worst at a rotation of 45°. However, at the same time the performance is not much worse to SIFT. The strong performance decrease of SIFT can be explained by the fact that the small scale key-points are lost because of the smoothing effect after the bilinear transformation.

Second, scale invariance is tested. As a reference image we used a down scaled image (0.8) in order to have scale changes in both directions. Figure 4(a) shows that our approach which passes on detecting key-points with small scales performs slightly better than SIFT.

Finally, we examined the repeatability of the detector and the matching of the descriptor by generating different projective transformations of the image.

[1] Available at http://www.cs.ubc.ca/ lowe/keypoints/

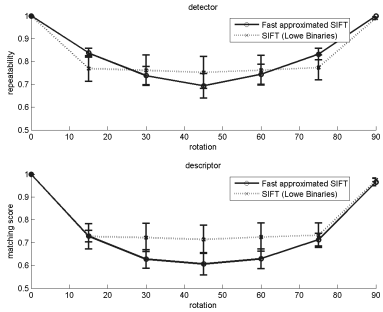
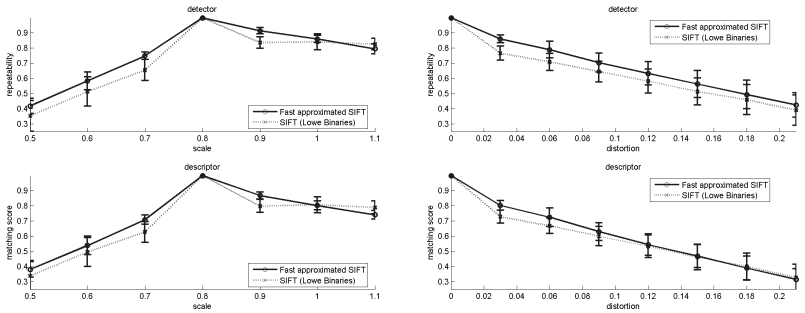


Fig. 3. Even in the worst case of a rotation of 45 °, approximated SIFT shows only a slight decrease in performance of the detector (above) and the descriptor (lower)



(a) Scale transformations (b) Projective transformations

Fig. 4. The proposed approximations of our method do not have any effects on scale or projective invariance

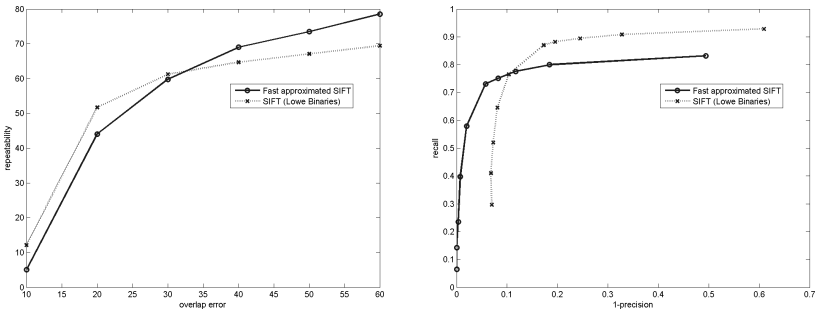
Again the results in Figure 4(b) show good performance for the approximated SIFT implementation.

4.2 Mikolajzyk Framework

We compared our method to Lowe's approach using the recently proposed framework from Mikolajzyk [9]. Two images of the Graffiti data-set have been used. The repeatability of both detectors are shown in Figure 5. When the overlap error tolerance is large enough the approximated SIFT implementation performs even better than the original version. However, allowing only a small overlap error, the approximation effects can be seen which lead to a slightly decreased performance. In Figure 5(b) we see a similar result for the descriptor.

4.3 Speed

We have a non optimized C++ implementation of the approximated SIFT which has been compared to the SIFT binaries provided by Lowe. In Table 3 the pro-



(a) Repeatability of the detector

(b) Descriptor evaluation using the ratio-nearest-neighbor matching criterion

Fig. 5. Evaluation results with the framework from Mikolajzyk

Table 3. Comparison of speed with respect to the image size

image size	SIFT (Binaries)	Approx. SIFT
800x640	4.24 s	0.625 s
400x320	1.34 s	0.180 s
200x160	0.44 s	0.075 s

cessing times for feature detection of different image sizes are listed. This experiment was done on a Pentium 4 with 3.2 GHz. Results show that approximated SIFT provides a speed-up of a factor 8 with this non optimized implementation where the major benefit is obtained in the detection process. Optimizing the implementation we expect to achieve at least a factor 12 to 16.

5 Conclusion

In this paper we have presented a novel approximation of the SIFT method that achieves a considerable speed-up of the original method (at least a factor of eight using our non optimized C++ implementation) while at the same time achieving comparable matching performance. We have carefully analyzed the speed-up gain theoretically and have performed extensive experimental evaluations.

This new fast SIFT variant opens several venues of further research which we are currently investigating. Once we have calculated the integral images the costs for the descriptor calculation is negligible. Therefore, we can perform a local neighbor search around a key-point for more discriminative/reliable descriptors. This should further increase the matching performance. Having such a fast method, tracking using SIFT becomes feasible. This should result in highly robust trackers. Another idea that is currently investigated is to use SIFT in an Adaboost framework. This has already been proposed by Zhang et al. [16], but having a fast SIFT will considerably speed-up the training process.

References

1. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
2. Mikolajczyk, K., Schmid, C.: Indexing based on scale invariant interest points. In: Proc. ICCV, Vancouver, Canada (2001) 525–531
3. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. In: Proc. CVPR. Number 1 (2004) 63–86
4. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: Proc. BMVC. (2002)
5. Se, S., Lowe, D., Little, J.: Local and global localization for mobile robots using visual landmarks. In: Proc. IROS. Volume 2. (2001) 414–420
6. Ke, Y., Sukthankar, R., Huston, L.: An efficient parts-based near-duplicate and sub-image retrieval system. In: Proc. Int. Conf. on Multimedia. (2004) 869–876
7. Tuytelaars, T., Gool, L.V.: Content-based image retrieval based on local affinely invariant regions. In: Proc. Int. Conf. on Visual Information and Information Systems. (1999) 493–500
8. Brown, M., Lowe, D.: Recognising panoramas. In: Proc. ICCV. (2003) 1218–1225
9. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. PAMI **27** (2005) 1615–1630
10. Ke, Y., Sukthankar, R.: PCA-SIFT: A more distinctive representation for local image descriptors. In: Proc. CVPR. Volume 2. (2004) 506–513
11. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. CVPR. Volume I. (2001) 511–518
12. Simard, P., Bottou, L., Haffner, P., LeCun, Y.: Boxlets: A fast convolution algorithm for signal processing and neural networks. In: Proc. NIPS. (1998) 571–577
13. Geusebroek, J., Smeulders, A., van de Weijer, J.: Fast anisotropic Gauss filtering. In: Proc. ECCV. (2002) 99–112
14. Lowe, D.: Object recognition from local scale-invariant features. In: Proc. ICCV. Volume 2. (1999) 1150–1157
15. Porikli, F.: Integral histogram: A fast way to extract histograms in cartesian spaces. In: Proc. CVPR. Volume 1. (2005) 829–836
16. Zhang, W., Yu, B., Zelinsky, G., Samaras, D.: Object class recognition using multiple layer boosting with heterogeneous features. In: Proc. CVPR. Volume 2. (2005) 323–330